Übung: Funktionen und Typen 2

- Algebraische Typen und Pattern-Matching
- Listen

1 Aufgabe

Das Ziel dieser Aufgabe ist es "Tools" zu entwickeln für ein simples Dateisystem. Jede der Teilaufgaben bringt uns diesem Ziel einen Schritt näher. Bauen Sie das Skript `Filesystem-todo.hs` schrittweise aus, bis Sie ein lauffähiges Programm haben. Sie müssen dazu alle `error "fixme"` durch Code ersetzen.

(a) Ein Dateisystem besteht aus Ordnern und Dateien, wobei Dateien einen Namen und eine Grösse haben. Studieren Sie `data FileSystem`, insbesondere die Konstruktoren, und implementieren Sie den verallgemeinerten Fold `filesystem`.

(b) Implementieren Sie `size` (unter Zuhilfenahme von `filesystem`).

(c) Implementieren Sie `existsFile` (unter Zuhilfenahme von `filesystem`).

(d) Studieren Sie den Typ `Path`. Es ist eine Darstellung von einem Pfad. Implementieren Sie die Funktion `findAll`, die alle Pfade zu einem Dateinamen findet.

2 Aufgabe (Bonus)

Das Ziel dieser Aufgabe ist es einen kleinen eigenen Interpreter für reguläre Ausdrücke zu schreiben. Bauen Sie dazu das Skript `Regex-todo.hs` schrittweise aus, bis Sie ein lauffähiges Programm haben. Sie müssen dazu alle `error "fixme"` durch Code ersetzen.

(a) Die Syntax von Regulären Ausdrücken ist wie folgt gegeben:

- Die Sonderzeichen ϵ (Epsilon) und $\varnothing$ (Empty) sind reguläre Ausdrücke.
- Jeder Char ist auch ein regulärer Ausdruck.
- Ist R ein regulärer Ausdruck, dann ist auch (R^*) ein regulärer Ausdruck.
- Sind R, S reguläre Ausdrücke, dann ist auch (RS) ein regulärer Ausdruck.
- Sind R, S reguläre Ausdrücke, dann ist auch $(R|S)$ ein regulärer Ausdruck.

(b) Die Semantik/Interpretation von regulären Ausdrücken ist dadurch gegeben, dass jeder reguläre Ausdruck einen gegebenen String gemäss den folgenden Regeln "matchen" kann oder nicht (vgl. Vorlesung Theoretische Informatik).

- Der reguläre Ausdruck $\varnothing$ matched keinen String.
- Der reguläre Ausdruck ϵ matched genau den leeren String (`""`).

- Ein regulärer Ausdruck von der Form `'c'` matched genau sich selbst als String (`"c"`).
- Ein regulärer Ausdruck von der Form (R^*) matched den leeren String sowie alle Strings, die man als Konkatenation von Strings, die R matchen konstruieren kann.
- Ein regulärer Ausdruck von der Form (RS) matched alle Strings, die man als Konkatenation von einem String, der R matched und einem String, der S matched konstruieren kann.
- Ein regulärer Ausdruck von der Form $(R|S)$ matched alle Strings, die entweder R oder S matchen.